Introdução ao LaTeX

Seminário LaTeX – o Livro

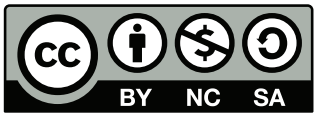
Geraldo Xexéo[1,2]

[1]Departamento de Ciências da Computação – IM/UFRJ
[2]Programa de Engenharia de Sistemas e Computação – COPPE/UFRJ

21 de maio de 2020 11:24

Geraldo Xexéo

Contato com o autor pelo e-mail xexeo@cos.ufrj.br.



Este arquivo e todos os arquivos relacionados podem ser encontrados em `https://github.com/xexeo/Seminario-LaTeX-2020`

Sumário

Agradecimentos

A enorme comunidade que desenvolve, compartilha ideias, relata bugs, escreve livros e pergunta e responde dúvidas sobre LaTeX na internet, meu sincero agradecimento por manter vivo um produto fantástico, mesmo que exótico para o mundo atual de interfaces fáceis e amigáveis.

Resumo

Esse texto é uma introdução ao LaTeX criada como resultado de um seminário de introdução ao LaTeX realizado na época do Covid-19, como uma ação extra classe e parte do 3º Seminário LUDES/LINE.

Foi escrito a partir dos slides do seminário que foi dado via Google Meet, de forma aberta no Programa de Engenharia de Sistemas e Computação da COPPE/UFRJ, tendo como foco os alunos do Programa. Porém, não deve ser muito diferente para qualquer aluno de áreas técnicas, como Física, Matemática, Econometria, etc.

Ele não pretende ser a melhor introdução ao LaTeX, mas pretende ser uma fácil e que faça o aluno entender um pouco o que está acontecendo quando o usa. Foi criada com a intenção de facilitar um pouco a vida dos meus alunos.

Todo os arquivos originais e pdfs finais do livro e do seminário estão disponível em: `https://github.com/xexeo/Seminario-LaTeX-2020`

Minha Motivação

Em 1990 eu comecei a usar o LaTeX no CERN. Lá eu tinha a meu dispor uma workstation MIPS Unix razoavelmente poderosa, pelo simples fato de ser o único a saber usar Unix no grupo. Meu uso da workstation fez crescer o interesse e o grupo acabou mudando de workstations VMS para workstations Unix HP. Parte da minha tese, um artigo e um exame de qualificação foram desenvolvidos naquele ambiente.

Tinha saído do Brasil acostumado com o Word 2.0 para DOS, e lá também usava também o Word para Mac no único microcomputador disponível para o grupo em que trabalhava. Isso não era problema para mim, sempre fui um híbrido. Na época usava DOS e Windows 3.1 no Brasil, Mac, Unix, e VMS no CERN, tendo acesso também a terminais IBM. Escolher a ferramenta certa era o melhor caminho para atingir uma solução. Mas seu trabalho não pode ficar dependendo de acesso a um computador compartilhado.

Então, para poder escrever nas workstations, fui aprender, sozinho, LaTeX. Sua saída me chamou atenção e aprendi a usá-lo com bastante confiança.

Ao voltar ao Brasil em 1993 e comprar um computador pessoal, voltei para o mundo DOS/Windows, já que o Linux não era uma realidade plena, e a facilidade do Word. Nele fiquei muitos anos, apenas com visitas esporádicas ao mundo do LaTeX.

Recentemente, porém, uma mudança ocorreu. Primeiro, passei a trabalhar com vários alunos que adotam o software livre como princípio. Obviamente que desprezam o uso de Word. Porém, suas alternativas abertas não atendem as necessidades do mundo acadêmico, ou mesmo as necessidades básicas de produção de texto em larga escala, devido a *bugs* ou comportamentos não esperados.

Ao mesmo tempo, apareceu no mercado um sistema que colocava todo o LaTeX disponível na nuvem, o ShareLaTeX , ainda permitindo a edição colaborativa. Rapidamente nos apropriamos desse sistema e passamos a usá-lo

para nossa produção. Ao mesmo tempo, o ShareLaTeX, agora comprado pelo Overleaf, fez renascer a comunidade da UFRJ que usa LaTeX, já que elimina a necessidade de manter o software atualizado e torna a edição independente do local onde está o aluno ou professor.

Com esse novo interesse, novos alunos passaram a precisar de entender como o LaTeX, enquanto alguns professores precisavam entender o que seus colegas ou alunos estavam fazendo. Isso me levou a dar o seminário e agora a escrever este texto.

Capítulo 1

Introdução ao Processamento de Texto

Os computadores foram criados para fazer contas, mas, na verdade, eles manipulam apenas símbolos. Rapidamente seus usuários perceberam que podiam ser usados para manipular textos e formatá-los de alguma forma para impressão. O processamento de texto, de forma geral, é possivelmente a área de software com o maior número de usuários, já que todo usuário de computador, alguma vez, escreveu algo e enviou para alguém, nem que seja um simples e-mail.

Para isso, foram criados vários tipos de programas que cumprem funções como a edição de arquivos de texto, a visualização de textos formatados, a impressão de forma adequada, e ainda processamentos adicionais, descritas na figura 1.1, criando o que pode ser chamado de uma cadeia de processamento de texto.

Cada sistema que trabalha com texto faz, prioritariamente, pelo menos uma dessas funções, agregando outras para fornecer algo mais sofisticado ao autor[1].

[1]Observa-se que esta cadeia pode, para outros contextos, ser representada de forma mais complexa, incluindo conceito de produção, versionamento, seguração, etc.

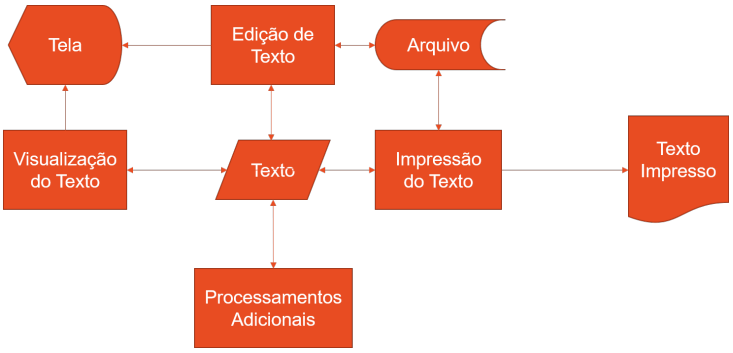


Figura 1.1: A cadeia de processamento de texto

1.1 Tipos de Sistema de Processamento de Texto

Dentro dessa cadeia de processamento existem vários tipos de programas. Os principais são:

- Editores de Texto
- IDEs
- Processadores de Texto
- Sistemas de Autoria
- Sistemas de Publicação (*Desktop Publishing*)
- Sistemas de Composição
- Sistemas Colaborativos de Edição

1.1.1 Editores de Texto

Editores de texto são programas que permitem ao usuário editar arquivos que são de **texto puro**. Texto puro é um conceito que mudou. Inicialmente significava que havia um mapeamento um para um entre o que você encontrava no arquivo, uma sequência de caracteres codificados em ASCII[2]. Em ASCII, por exemplo, a letra "A" é representada pelos bits "1000001" e a "a" por "1100001". Atualmente são usadas codificações que permitem que um caracter ou símbolo seja representado por uma sequência mais longa de bits,

[2]ASCII é um padrão que associa uma letra, e outros símbolos usados em arquivos, a um byte. Seu nome significa American Standard Code for Information Interchange. Baseado no alfabeto inglês, originalmente usava apenas 128 símbolos (7 bits), não possuindo os caracteres acentuados de outras línguas.

por meio de *escape codes*, por exemplo UTF-8[3], o que significa que os editores de texto, normalmente, não representam mais perfeitamente o arquivo em disco. Os sistemas de codificação atuais são normalmente extensões do ASCII, isto é, os 128 códigos de 1 byte do ASCII original ainda são válidos. A figura 1.2 mostra a função de um editor de texto na cadeia de processamento de texto.

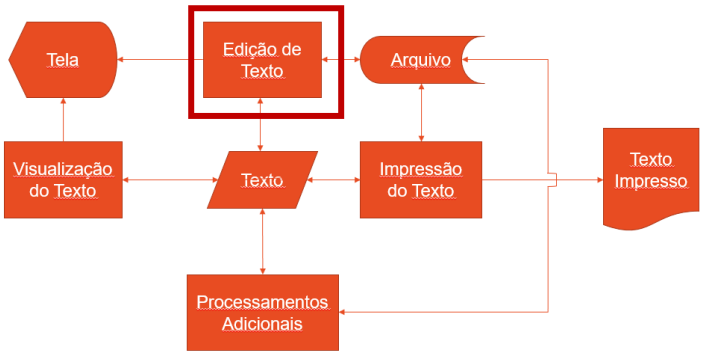


Figura 1.2: Função de um editor de texto na cadeia de processamento de texto.

Editores de texto foram necessários assim que trocamos as entradas por cartão e fita, que eram editados em máquinas não conectadas ao computador, por terminais ligados diretamente aos mesmos. Os primeiros editavam linha a linha, a seguir outros exigiam que o usuário gerenciasse o *buffer*, o seja, a parte do arquivo que estava em memória. Com o tempo chegamos a versões semelhantes as atuais.

Atualmente editores de texto têm um conjunto complexo de funções de busca, substituição, etc.

Exemplos de editores de texto atuais são o Notepad, que vem por *default* com o Windows, o `vi` e o `vim`, Notepad++, EditPlus, TextEdit e o poderosíssimo Emacs. A figura 1.3 mostra um exemplo to `vim`.

1.1.2 IDEs

Uma IDE, ou "Integrated Development Environment", é uma extensão lógica da ideia de editor de texto criada originalmente para programadores, fornecendo serviços adicionais a edição de texto, como compilação, *debugging*,

[3]Unicode Transformation Format, que permite codificar 1.112.064 símbolos

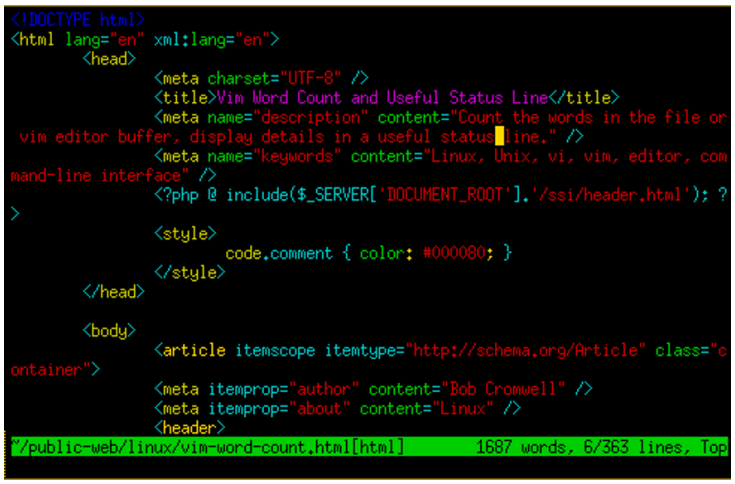


Figura 1.3: O editor de texto vim

controle de versão, normalmente por meio de interfaces com os programas que fazem isso.

Uma IDE cobre a parte de edição e processamento de texto da cadeia de processamento de texto, como visto na figura 1.4.

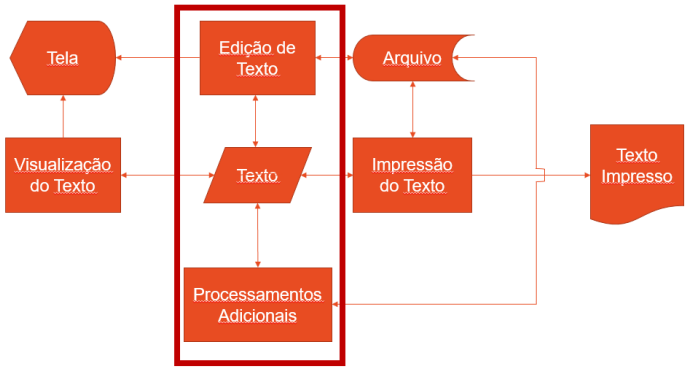


Figura 1.4: Um IDE permite editar e invocar outros processsamentos de texto

Exemplos de IDE são o Atom, o Visual Studio, o TEXStudio e o próprio Emacs. Com a evolução dos editores de texto, a fronteira entre editores e IDEs ficou indefinida, muitas vezes dependendo do uso que o usuário faz do programa. A figura 1.5 mostra um exemplo do TEXStudio.

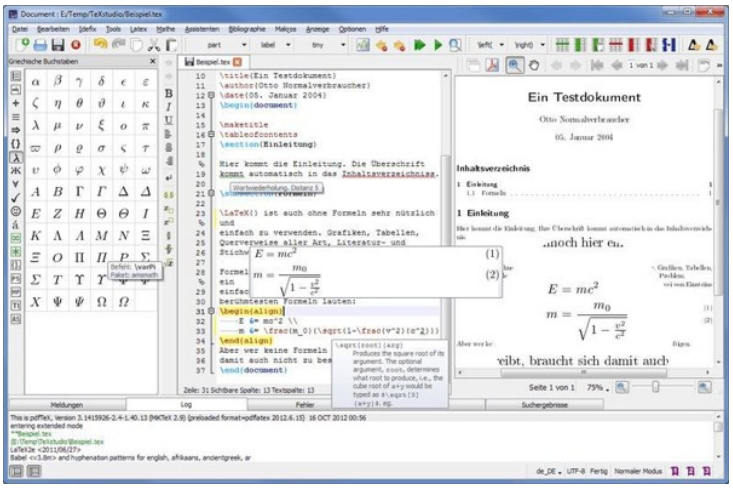


Figura 1.5: O TEX Studio

1.1.3 Sistemas de Composição (*Typesetting*)

Já que era possível editar textos, porque não imprimi-los de forma adequada? Essa ideia levou a criação de programas de tipografia, que faziam a tradução de um arquivo texto, com marcações adequadas, para um outro arquivo que fosse interpretado em uma impressora (ou outras máquinas mais sofisticadas de *typesetting* digital).

Um sistema de tipografia cobre apenas a parte de impressão da cadeia de processamento de texto, como visto na figura 1.6.

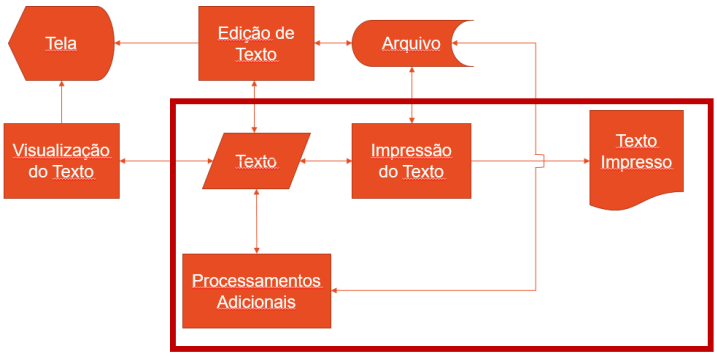


Figura 1.6: Sistemas de tipografia imprimem e fazem processamentos adicionais.

Essas marcações adequadas são como comandos, o que leva ao conceito de linguagem de marcação, tratado na próxima seção. Um arquivo de texto

marcado se assemelha a um programa de computador, por possuir palavras código que dão os comandos necessários, e o processo de transformá-los em um arquivo a ser impresso é uma compilação. Sistemas desse tipo não possuem editores associados e são normalmente ativados por linha de comando.

Exemplos de sistemas de tipografia são o troff, o TEX e o LATEX. A figura 1.7 mostra o LATEX sendo usado em linha de comando.

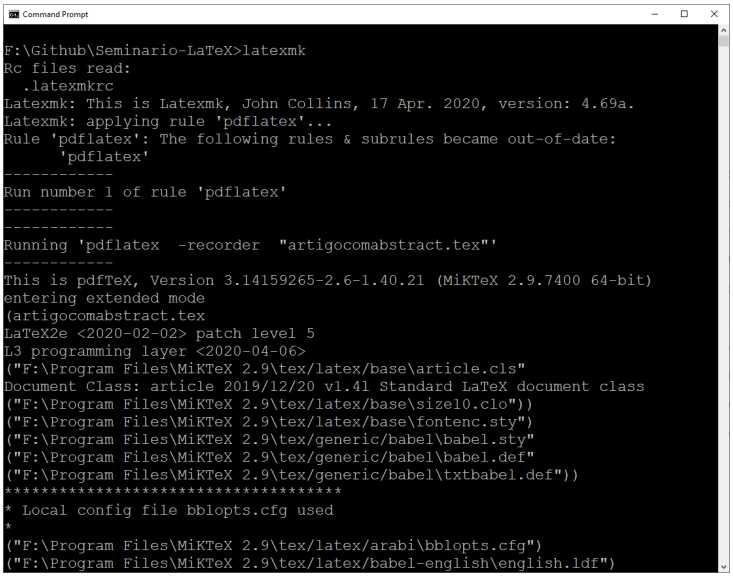


Figura 1.7: O LATEXsendo usado

Os sistemas de composição mais avançados, como o LATEX, ou sistemas baseados em SGML, como o próprio HTML, tem a característica de separar a lógica do texto, sua estruturação, da sua forma de apresentação. Isso é conseguido no LATEX por meio do uso de classes de documentos, pacotes e comandos criados pelo usuário com essa intenção. No HTML isso é conseguido por meio do uso de classes para as tags e especificações CSS que definem a aparência das tags que pertencem aquela classe. Em XML é feito por meio de um processador externo que usa uma especificação que diz como o XML deve ser apresentado, por exemplo usando XSL como linguagem para definir o estilo a ser usado na apresentação.

1.1.4 Processadores de Texto

Em certo ponto, ficou claro que uma das principais utilidades do computador seria permitir a criação de textos a serem publicados, logo um editor de texto

deveria ser estendido para suportar funções como colocar palavras em negrito, itálico, selecionar fontes, etc. Com o advento do micro-computador, esse se tornou o tipo de programa mais usado.

Processadores de texto cumprem grande parte das funções na cadeia de processamento de texto, como mostra a figura 1.8.

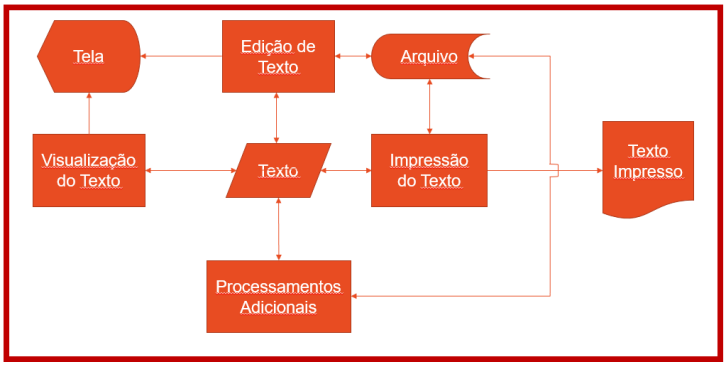


Figura 1.8: Os processadores de texto na cadeia de processamento de texto.

Com a evolução dos terminais de computadores, micro-computadores e placas de vídeo e monitores, os processadores de texto evoluíram de sistemas que mostravam alguma coisa do que estava sendo prevista para o texto, como caracteres em bold, para sistemas que permitiam uma visualização prévia, até chegar a edição pelo conceito de WYSIWYG, *What You See Is What You Get*, lido "uiziwig", que mostra na tela quase que exatamente o que será visto na edição final, sendo as diferenças mínimas e quase imperceptíveis causadas por questões tecnológicas e da diferença entre papel e tintas e monitores.

Atualmente o Word é o processador de texto que domina amplamente o mercado, e seus principais concorrentes são sistemas de código aberto, como o LibreOffice Writer ou o Apache OpenOffice Writer. Um tela do Word é mostrada na figura 1.9.

Uma característica avançada importante dos processadores de texto é o uso de estilos e de templates, que são basicamente coleções de estilos organizadas de modo a dar um formato coerente ao documento. Por meio do uso de estilos é possível garantir que partes do documento que tem a mesma função tenham a mesma aparência.

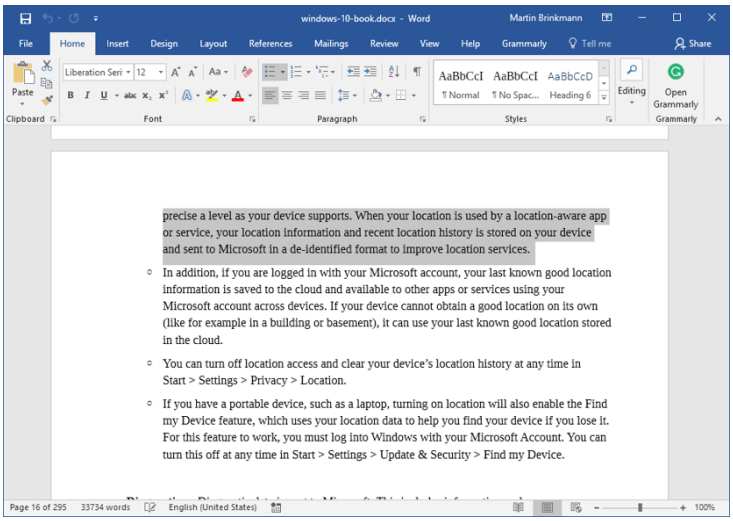


Figura 1.9: O Word é o principal processador de texto do mercado.

1.1.5 Sistemas de Autoria

Sistemas de autoria são uma evolução interessante dos processadores de texto, ou dos IDEs, voltadas para autores de livros, roteiros, etc. que não só permitem editar o texto, em formatos específicos, como também guardar informações como fichas de personagens, descrições de cena, *storyboards*, etc... Eles podem cobrir toda a cadeia de processamento de texto.

Mesmo o mercado não sinalizando isso, sistemas de autoria tem grande potencial para o futuro e para a academia, pois normalmente ao escrever algo precisamos guardar muito informação relativa ao conteúdo, e um sistema de autoria poderia ajudar a fazê-lo.

Exemplo de sistemas de autoria são o Scrivener, apresentado na figura 1.10, o Final Draft e o Celtx.

1.1.6 Sistemas de Publicação

Sistemas de publicação, ou *Desktop Publishing*, são sistemas cujo o foco é a diagramação de publicações, fornecendo uma capacidade menor de processamento de texto. São nesses sistemas que a maioria de jornais e revistas, e tamém manuais, folhetos, e outras formas de documentos impressos, são hoje preparados para a impressão.

Exemplos típicos de sistemas de publicação são o Publisher e o Frame-

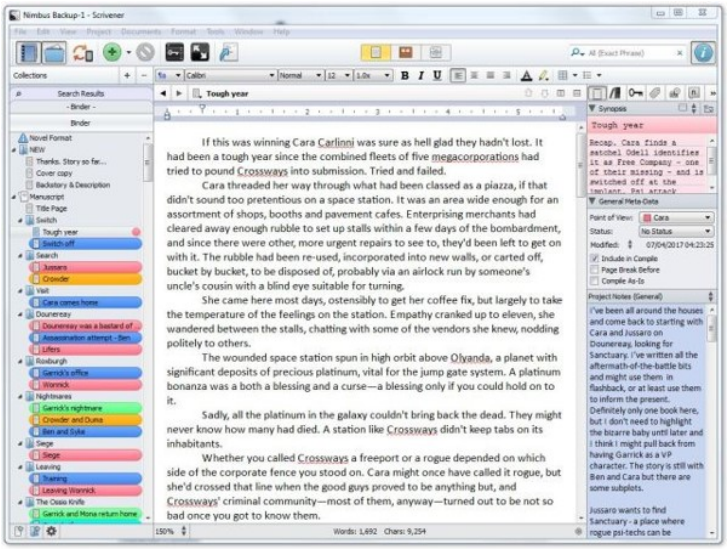


Figura 1.10: Uma tela do Scrivener

maker, que é mostrado na figura1.11.

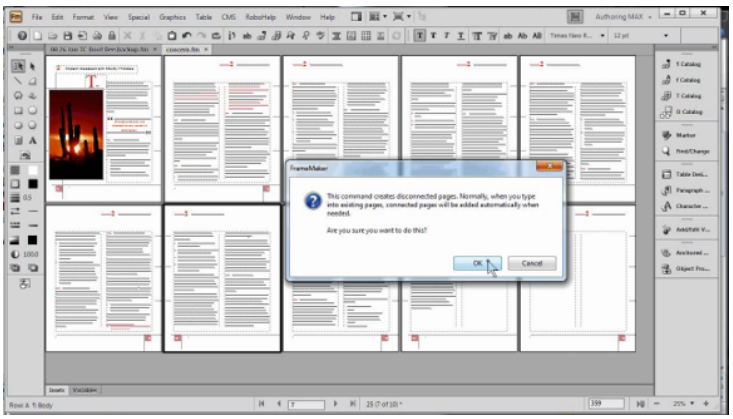


Figura 1.11: O software Framemaker

Sistemas de publicação são mais difíceis de usar que sistemas de processamento de texto, porém permitem fazer a diagramação de um documento de formas mais avançadas, e depois colocar os textos nos espaços dentro da diagramação. Normalmente o texto é editado em outros programas, por exemplo, em um processador de texto, que permite colocar já os formatos como negrito e itálico.

1.1.7 Sistemas Colaborativos de Edição

Esses sistemas aparecem cedo, porém se expandem principalmente com o fortalecimento da internet. Sua função básica é permitir que mais de uma pessoa edite um arquivo ao mesmo tempo, mas é importante que uma pessoa não atrapalhe a outra e também que haja alguma percepção do que o outro está fazendo. Atualmente o mais conhecido é o Google Docs, que é um processador de texto limitado em funcionalidade mas muito fácil de usar.

O ShareLateX foi um sistema colaborativo de edição voltado para o LaTeX que acabou sendo responsável por um renascimento do uso do LaTeX na academia. Acabou sendo incorporado ao concorrente Overleaf, cuja interface aparece na figura 1.12.

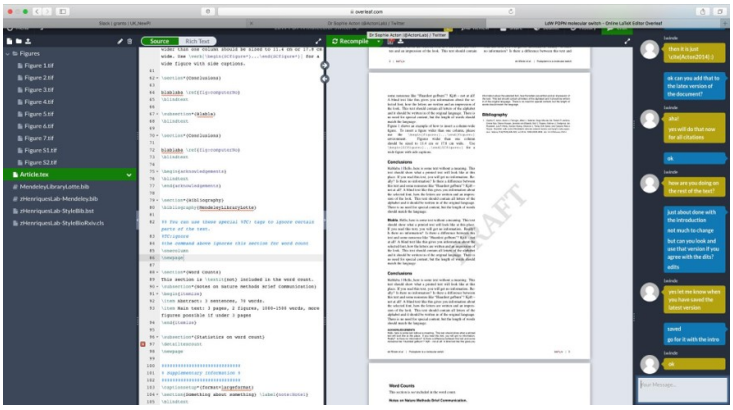


Figura 1.12: O Overleaf.

1.2 Tipos de Linguagens para Arquivos de Texto

Arquivos de texto podem ser guardados de várias formas. No limite, podemos fotografar uma página de texto e guardar a imagem, mas geralmente queremos um arquivo que possa ter seu texto manipulado.

Normalmente, na cadeia de processamento de texto, o arquivo é lido e colocado em memória em um formato mais adequado, e depois, quando salvo, é "rearrumado" de alguma forma, um formato de arquivo texto. Por exemplo, ao salvar um arquivo Word você pode escolher o seu formato nativo (.docx)

ou vários outros formatos alternativos, como RTF[4], ou mesmo um arquivo texto, nesse caso perdendo toda a formatação.

Os formatos de arquivo podem ser divididos em:

- Linguagens de Impressão/Visualização
  - PostScript, DVI, PDF
- Linguagens Intermediárias (de impressão)
  - .dvi
- Linguagens de Marcação
  - SGML, HTML, TEX LATEX Markdown, RPF, fods

O fato de uma linguagem de marcação ser legível por humanos não quer dizer que seja facilmente legível, principalmente quando geradas por máquinas. A figura 1.13 mostra um exemplo de arquivo Postscript.

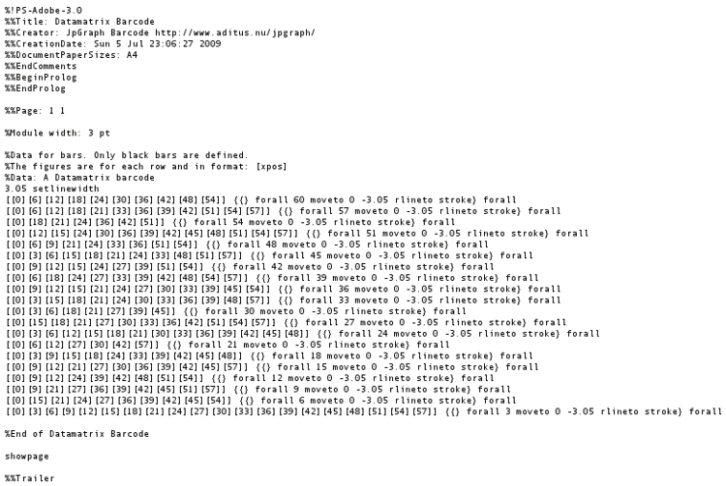


Figura 1.13: Exemplo de um arquivo PostScript

1.2.1 Linguagens de Marcação

Um linguagem de marcação é caracterizada por um conjunto de códigos que é aplicado sobre o texto, ou sobre qualquer forma de dados, com o fim de adicionar informações específicas sobre esse texto, a cada trecho específico, como por exemplo a forma de exibi-lo graficamente.

Linguagens de marcação são uma forma simples de indicar como um texto deve ser impresso, e são usadas desde o início da digitalização do processo de composição e impressão.

[4]Rich Text Format

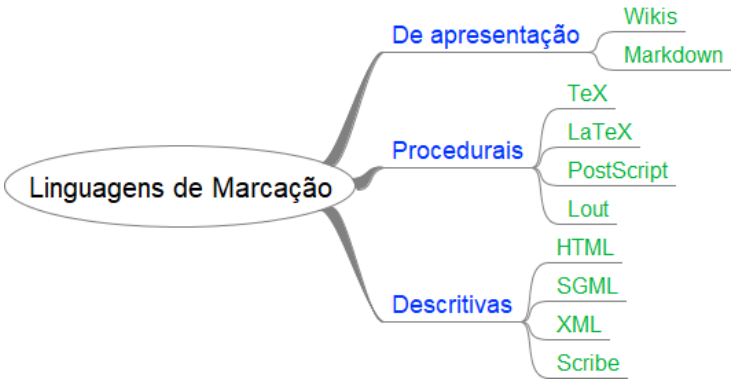


Figura 1.14: Mapa mental das linguagens de marcação.

As linguagens de marcação padronizadas permitem que programas diversos cumpram a mesma função. Elas podem ser divididas em:

- Procedurais
  - troff, TEX , LATEX, Postcript
- De apresentação
  - Wikis, Markdown
- Descritivas
  - SGML, HTML, XML

Linguagens de marcação procedurais normalmente incluem instruções de composição, ou seja, elas mantém unificadas a estrutura e a apresentação, enquanto linguagens de apresentação não se preocupam com nada além da apresentação imediata. Já linguagens descritivas, introduzidas por Scribe, separam a estrutura da apresentação. LATEX, apesar de ainda conter instruções de composição, foi fortemente influenciada por essa ideia, e o usuário final praticamente só usa instruções que falam sobre a estrutura do documento. Um mapa mental das linguagens pode ser visto na figura 1.14(Adams, 2007).

Linguagens de marcação normalmente são criadas para serem simultaneamente compreensíveis para o ser humano e tratáveis por um programa de computador.

Um exemplo de código LATEX e seu resultado é mostrado na figura 1.15.

1.2.2 Linguagens de Impressão

Linguagens de impressão tem como finalidade servir apenas para o processamento por um mecanismo de impressão ou um software específico de visua-

```
1   \textbf{Um exemplo}
```

Um exemplo

Figura 1.15: Exemplo de código LaTeX

lização. Nesse caso, não há preocupação com a comprensão do formato por seres humanos.

Exemplos de linguagens de impressão são o formato PDF, que é um Postscript empacotado de forma a descrever documentos compostos de páginas.

TeXtrouxe o conceito de uma linguagem de impressão intermediária, por meio do formato DVI, que torna o arquivo gerado pelo TeX independente do processamento final, o que permite que cada fabricante, ou interessado em geral, faça um programa próprio de conversão de DIV para qualquer formato de impressão. Logo, documentos DVI podem ser transformados em PDF, PS ou outro formato por software especiais, mas atualmente as implementações de LaTeXtornam isso transparente.

1.3 O que é o LaTeX

LaTeX(Lamport, 1994) é um sistema de composição, ou *typesetting*, baseado no TeX e apoiado com outros programas, como o biber e o latexmk. Ele permite usar arquivos de texto marcados para criar arquivos a serem impressos, possivelmente no formato PDF, seguindo regras de composição e usando fontes detalhadamente criadas para reproduzir a qualidade de fontes utilizada na composição manual, incluindo especialmente as ligaduras, que são caracteres especiais que representam dois ou mais caracteres de forma visualmente mais elegantes, como mostradas na figura 1.16.

1.4 Por que LaTeX?

Alguns argumentos dados para usar LaTeX são:

- qualidade estética, os defensores de LaTeX dizem que por ser um sistema de composição, baseado em algoritmos que buscam soluções ótimas segundo regras específicas e muitas vezes desconhecidas pelos autores;

fi	$\longrightarrow$	*fi*	*AE*	$\longrightarrow$	*Æ*
ff	$\longrightarrow$	*ff*	*ae*	$\longrightarrow$	*æ*
ffi	$\longrightarrow$	*ffi*	*OE*	$\longrightarrow$	*Œ*
fl	$\longrightarrow$	*fl*	*oe*	$\longrightarrow$	*œ*
ij	$\longrightarrow$	*ij*	*LATEX*	$\longrightarrow$	*LaTeX*

Figura 1.16: Exemplos de ligaturas de fontes do TEX.

- preço, LATEXé grátis;
- o documento .tex é independente de fabricante, pode ser editado com qualquer editor e ser usado plenamente com gerenciadores de versão. Você sempre poderá editar um arquivo .tex;
- torna fácil trocar o formato de arquivos grandes, trocando apenas o estilo;
- WYSIWYG é na verdade um What You See Is All You've Got, WYSIAYG, não sendo possível modificar o comportamento além do que o sistema está oferece para você[5];
- foco no conteúdo, deixando as questões de formato para os criadores de estilo, e
- longevidade, está em utilização ampla há mais anos, sem mudanças de comportamento.

Alguns problemas de LATEXsão:

- esforço de tempo para aprender LATEX;
- dificuldade de resolver alguns erros;
- algumas interações indesejadas entre pacotes;
- dificuldade de criar um estilo novo como você deseja, e
- as alterações não aparecem imediatamente, tendo que ser compiladas.

1.4.1 O Design Lógico de Documentos

A ideia do design lógico de documentos, proposta inicialmente por Reid (1980), defende que o autor não deve se preocupar com o formato enquanto escreve o documento. Isso é exatamente o oposto que os sistemas WYSIWIG induzem, já que sempre estamos vendo o que seria o formato final.

Essa ideia influenciou certamente o uso de **estilos** em várias outros pro-

[5]Isso fica mais claro se você usar um sistema WYSIWYG mais limitado como o Google Docs

gramas e linguagens. Um estilo, de forma muito geral, é uma definição de como um pedaço do texto deve ser impresso, de forma que possa ser aplicado em muitos lugares no texto em si, por meio de alguma regra de associação.

No Word nós normalmente utilizamos os estilos "Heading 1", "Heading 2", etc, enquanto no LaTeXusamos os estilos `\section{}`, por exemplo. Todas essas formas de estilo podem ter várias propriedades alteradas, como o tamanho do fonte, se é negrito ou não, e essa mudança acontece em todas as partes do texto associadas a elas. Isso normalmente é feito, em um arquivo, por meio de uma linguagem de marcação.

Um conceito semelhante, com um grau a mais de indireção, são as linguagens que permitem marcar o texto de uma forma e associar a marcação o estilo desejado. Isso acontece em HTML 5, onde o arquivo HTML pode ser todo marcado de uma forma, mas o estilo CSS é dividido em classes que podem ser associadas a vários tags diferentes do HTML, por meio da chave `style=`. Dessa forma o design lógico do documento fica independente do desing visual do documento.

1.5 Sistemas Mais Usados na Computação

Nesta seção capítulo vamos ainda comparar mais detalhadamente três sistemas que são hoje muito usados na Computação:

- **Word**
  - Um software WYSIWYG
  - Líder do mercado
  - Superpoderoso
  - Difícil de usar para trabalho colaborativo
- **Google Docs**
  - Quase WYSIWYG
  - Sucesso entre os jovens
  - Pouca capacidade de plena expressão gráfica
  - Ótimo para trabalho colaborativo
- LaTeX
  - Melhor imagem de texto, mas no detalhe
  - Ótimo para Matemática
  - Difícil de usar
  - Empoderado pelo Overleaf
  - Renovado com os sistemas colaborativos

Desses sistemas, claramente o Google Doc ainda não serve para gerar

documentos com formatação de qualidade, ou documentos destinados a artigos de revistas ou livros. Então ficaremos, na próxima seção, apenas com a discussão dos mitos e fatos que aparecem na discussão do uso de Word vs LaTeX.

Chamamos a atenção que não consideramos viável o uso para objetivos acadêmicos dos sistemas abertos, como Open Office ou LibreOffice, por possuírem problemas de compatibilidade dos resultados de visualização entre versões dos próprios e com formatos disponíveis para o Word[6].

1.6 Mitos e Fatos de Word vs LaTeX

1.6.1 LaTeX é mais produtivo

A suposição seria que, como você não se preocupa com o resultado final, já que ele é controlado pela classe de documento e estilos. Isso é falso, e já foi comprovado em pesquisas científicas que a produtividade é igual ou menor.

1.6.2 A qualidade de saída do LaTeX é muito melhor

Verdade, mas possivelmente só para quem entende o que está vendo. A qualidade da saída proporcionada pelo TeX depende de algum conhecimento do que é uma boa fonte, uma boa distribuição de texto. É possível que um leitor tenha alguma sensação de beleza ou conforto superior quando vê um texto gerado pelo TeX , mas é possível também que não saiba diferenciar, pois são detalhes.

1.6.3 Word não trabalha bem com fórmulas matemáticas

Falso. A nova maneira de trabalhar com fórmulas do Word, que já está funcionando há algum tempo, não só é muito boa como permite escrever as fórmulas com a mesma sintaxe do que o LaTeX.

Ainda há a questão da beleza das fórmulas, mas a diferença também é pequena hoje em dia.

[6]Será isso um mito? Não é o que parece pela experiência do autor

1.6.4 Você perde muito tempo com besteira no LaTeX

Verdadeiro. Como LaTeX é uma linguagem compilada, você pode perder muito tempo com erros espúrios. Além disso, a capacidade de detectar o erro no momento certo é fraca no LaTeX, e coisas como parenteses faltando podem causar mensagens de erro muito longe. Porém, com o uso, isso tende a diminuir.

1.6.5 Não consigo colocar a imagem onde quero no Word

Falso. Sinceramente, nem sei de onde vem esse mito. Você pode colocar a imagem onde quiser no Word. O mesmo, com o LaTeX, já é mais díficil.

1.6.6 Em LaTeX gerencio melhor as bibliografias

Falso. Na verdade, LaTeX não gerencia a bibliografia, ele implica no uso de outro sistema, o BibTeX, por exemplo, e outro processador, como o biber. O mesmo pode ser feito com o Word, usando sistemas como Zotero ou Paper.

1.6.7 Em LaTeX consigo controlar versões

Verdadeiro. Este autor é um grande fã do uso de gerência de versões, sendo um usuário do Git e do GitHub. Realmente é fácil fazê-la com o LaTeX, porque todos os arquivos são textuais e plenamente controlados no controle de versões, porém um efeito quase similar para a visualização de diferenças pode ser obtido com a instalação do software `pandoc` e uma pequena configuração do Git. Outras funcionalidades, como o *merge* de versões com conflito não são possíveis.

1.6.8 Word é mais fácil de aprender

Verdade, porém o usuário médio não usa corretamente as funções que tornam o software Word um programa fantástico, como o uso de templates e estilos.

1.6.9 Com o Word, basta ele

Falso. Para fazer tudo que você faz com o LaTeX, você precisa de outros programas para o Word também.

1.6.10 Word tem problemas com arquivos grandes

Verdade, para arquivos muito grandes, o Word realmente às vezes se perde e causa bugs, quase impossíveis de consertar, com arquivos muito grandes. A alternativa de usar arquivos *master* que incluem outros arquivos, o que se faz com facilidade no LaTeX, está quebrada há várias versões e não há perspectiva de ser corrigida[7].

1.7 Recomendações

As três grandes opções disponíveis hoje em dia são: Word, Google Docs e LaTeX. Claro que existem vários outros programas que podem ser usados, como vários citados aqui. A questão é que, usando outros programas além desses três, há a possibilidade de problemas de compatibilidade com outros autores ou com a editora. Esses três programas cobrem realmente grande parte da população, mas existem pessoas que usam outros. Por exemplo, existe uma versão Markdown para fazer a tese na COPPE, que gera arquivos LaTeX. E alguns alunos insistem em usar software aberto como Open Office. Em todo caso, as principal escolha deve ser entre esses três programas. Seguem algumas considerações.

- **Word**
  - Ótima opção para um autor e um revisor
  - Bom para arquivos pequenos e grandes, não use para arquivos muito grandes.
  - Separe o texto em capítulos, e só junte na hora de imprimir.
  - Cuidado com o backup, faça sempre
  - Se possível, use o controle de versões
  - Aprenda a usar estilos, principalmente de parágrafos, e templates
  - Não separe parágrafos com linha em branco e não comece parágrafos com tab, use estilos para isso.
- **Google Docs**
  - Ótimo para colaborações onde pessoas escrevem ao mesmo tempo

[7] Sim, isso é algo bizarro de se saber.

  - Sem o controle de quem fez o que, o que é possível com controle de versão (opção *Blame* no GitHub)
  - Funciona bem com arquivos pequenos, não foi testado com arquivos grandes ou enormes
  - Alguns editores exigem arquivos Word.
- **LaTeX**
  - Se a editora fornece o formato (.sty) é a melhor opção
  - Para exames, dissertações e teses da Coppe é a melhor opção
    - Use o formato CoppeTeX disponíveis em: `https://github.com/COPPE-UFRJ/CoppeTeX`
    - basta baixar todo o diretório dist
  - Use o JabRef ou o Zotero com Better BibTeX for Zotero
  - LaTeX no Overleaft
    - Bom para artigos
    - Pode não conseguir compilar uma tese, pois tem limite de tempo.
    - Ligue as opções GitHub e Dropbox, e faça backup
  - LaTeX em casa
    - Use o Git+Nuvem (GitHub, GitLab, etc.)
    - Mantenha a instalação atualizada
    - Faça backup

Capítulo 2

O Mundo LaTeX

Nesse capítulo é feita uma introdução aos conceitos que giram em torno do LaTeX.

2.1 Invenção do LaTeX

No começo, não existiam computadores. Livros eram escritos a mão ou datilografados e um tipógrafo os montavam por meio de tipos móveis, página a página, para imprimi-los, primeiro como um grande carimbo como as máquinas de Gutenberg, depois por processos mais sofisticados, como o linotipo ou o offset.

Ao escrever o primeiro volume da série seminal de livros *The Art of Computer Programming*, Donald Knuth, que ganhou o prêmio Turing de 1974, teve o livro feito da forma clássica, por composição a quente. No segundo volume, em 1969. foi utilizada uma composição digital. Seu desagrado com o resultado visual o levou a um projeto de 10 anos em tipografia digital, período em que criou o TeX o METAFONT, um método de programação conhecido como *Literate Programming* e os programas `WEB` e `CWEB` que o implementam.

Um texto processado em TeXaparece na listagem 2.1, e seu resultado na figura 2.1.

Listagem 2.1: Exemplo de arquivo em TeX puro.

```
1 % exemplo de comentário
2 % Introdução ao LaTeX
3 % Seminário LaTeX -- o Livro
4 % Geraldo Xexéo
```

```
% Este arquivo tem a licença Creative Commons
% BY-NC-SA 2020

\magnification\magstep1
\font\logo=logo10 % font used for the METAFONT logo
\def\MF{{\logo META}\-{\logo FONT}}
\def\TeX{T\hbox{\hskip-.1667em\lower.424ex\hbox{E}
\hskip-.125em X}}
\def\bib{\par\noindent\hangindent 20pt}
\line{\bf Um exemplo de arquivo \TeX\ \ \ \hfill}
\bigskip\noindent
Este texto é um exemplo simples de como pode ser
gerado um arquivo \TeX.
A linguagem permite
usar comando e criar macros, tendo sido usada para escrever
{\sl The Art of Computer Programming}.
\bye
```

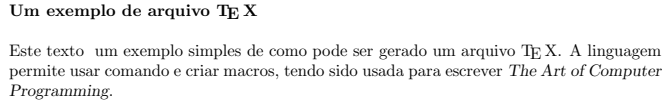


Figura 2.1: Resultado do exemplo do uso do TEXpuro.

Deve ficar claro que a linguagem TEX é uma linguagem de marcação orientada a como vai ser a visualização do texto. Já na década de 80, influenciado pelas propostas de descrever um texto pela sua lógica proposta pelo sistema Scribe(Reid, 1980), separando a visualização, Leslie Lamport, que recebeu o prêmio Turing de 2013, escreveu algumas macros para si, que foram distribuídas para colegas, até que foi convidado a escrever um livro que as descrevesse(Mittelbach, Goossens et al., 1999). A versão atual do livro se chama *LaTeX(2nd Ed.): A Document Preparation System: Users Guide and Reference Manual*(Lamport, 1994).

É importante notar que os comportamentos previstos tantto do TEX quanto do LaTeX são bastante estáveis, porém o LaTeXsobre evolução constante, tanto de seu funcionamento básico como das centenas de pacotes dis-

poníveis na CTAN.

2.2 Situação Atual do LaTeX

As implementações mais conhecidas de LaTeXe os programas auxiliares são:

- Várias implementações de TeX
  - pdfTeX – processador que ficou sendo o mais usado
  - XeTeX – Unicode + novas fontes
  - LuaTeX – evolução do pdfTeX onde todos as chamadas internas podem ser acessadas por Lua
- Programs relacionados
  - BibTeX – programa original de tratamento de citações e bibliografia
  - biber – processador de referências mais moderno, para usar o bibLaTeX.
  - JabRef – gerenciador de referências
- Editores
  - TeXnicCenter
  - TeXStudio
  - TeXMaker

Diferentes distribuições incluem diferentes versões de TeX, LaTeX e os programas relacionados. As principais distribuições são:

- Multi-sistema
  - TeXLive – A principal distribuição, com dezenas de desenvolvedores
- No Windows
  - MiKTeX – Uma distribuição ativamente mantida e usando *wizards* para instalação, preferida por muitos usuários Windows por causa dessa facilidade
  - proTeXt – MiKTeX com alguns adicionais
- No Linux –
  - Pacote disponível na distribuição, geralmente uma versão do TeXLive, normalmente de atualização mais lenta.
- No Mac
  - MacTeX – TeXLive especializada para o Mac

Também é possível usar o LaTeXna nuvem, nos seguintes sites:

- Overleaf

    - `https://www.overleaf.com/`
    - O principal ambiente de edição compartilhada de LaTeX, principalmente depois de comprar o ShareLaTeX
- Papeeria
    - Permite editar LaTeXe Markdown, baseado no TeXLive
    - `https://papeeria.com/`
- Cocalc
    - `https://cocalc.com/doc/latex-editor.html`

As principais informações sobre LaTeXe TeX podem ser obtidas em:

- CTAN Comprehensive TEX Archive Network: https://ctan.org/
- TeX Stack Exchange : https://tex.stackexchange.com/
- TeX FAQ https://texfaq.org/
- TeX User Group TUG: https://www.tug.org/
- The LaTeX Project: https://www.latex-project.org/

2.3 Como funciona o LaTeX

Na prática, o LaTeX funciona como um compilador. Sua entrada principal é um arquivo .tex, que contém o texto do documento a ser impresso. Na verdade, vários arquivos são lidos, muitos instalados na distribuição e fora da visão imediata do autor, sendo os principais:

- um arquivo .cls, que define a classe do documento;
- zero ou mais arquivos .sty, que são as implementações dos pacotes utilizados;
- arquivos que definem como será tratada a bibliografia, como .bbx, .cbx ou .lbx, que são parte de pacotes de bibliografia e chamados pelo pacote;
- arquivos com a bibliografia propriamente tida, .bbl no caso do biber estar sendo usado;
- arquivos .tex incluídos pelo autor a partir do documento raiz, e
- arquivos de imagem, .png, .jpg, .pdf ou outros, incluídos pelo autor a partir do documento raiz.

Com o uso de citações e bibliografia, o LaTeX precisa fazer o processamento pelo menos 3 vezes: a primeira para gerar um arquivo de entrada para o biber, que é então executado, a segunda para colocar o que o biber gerou em seu lugar correto e a terceira para ajustar todas as referências de acordo com a configuração final do documento. Esse processo, que é apenas uma descrição parcial do que é mais visível ao autor, é representado na figura 2.2.

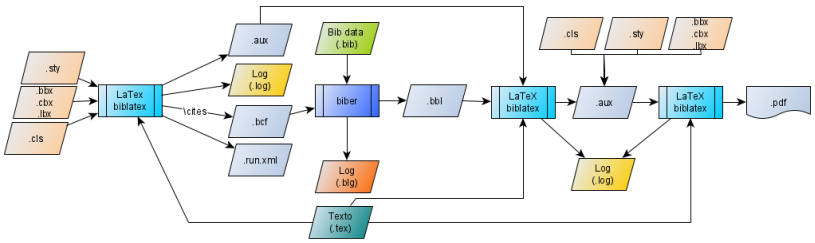


Figura 2.2: O fluxo de processamento (simplificado) do LaTeX.

2.4 Recomendações de uso

- Comece pelo Overleaf, até ter interesse de mudar para sua máquina
- Sincronize o Overleaf via Dropbox e GitHub.
- No PC
  - Comece com o MiKTeX, é muito mais fácil de instalar e manter atualizado;
- No Linux
  - Use o TeXLive em vez da distribuição padronizada
- No Mac
  - Aparentemente, sua única opção é o MacTeX.
- Use o TeX Studio para editar.
- Para processar, use o LuaTeX e LuaLaTeX
- Para bibliografia, use o BibLaTeX (forma do biber
  - Cuidado com estilos que exigem o BibTeX
  - Há diferenças sutis de comportamento
- Sempre mantenha seu projeto LaTeX no Git
  - GitHub, GitLab e etc.

Capítulo 3

LaTeX Básico

3.1 LaTeX Muito Básico

3.1.1 Documento Mínimo

Um documento mínimo LaTeX tem a aparência da listagem 3.1.

Listagem 3.1: Um documento mínimo em LaTeX.

```
1 \documentclass[a4paper]{article}
2 % Introdução ao LaTeX
3 % Seminário LaTeX -- o Livro
4 % Geraldo Xexéo
5 % Este arquivo tem a licença Creative Commons
6 % BY-NC-SA 2020
7
8 % arquivo semi1.tex
9 %isto é um comentário
10 %preâmbulo
11 %onde colocamos comandos
12 \begin{document}
13     %área de texto
14     Hello World!
15 \end{document}
```

Cada documento LaTeX começa com a declaração de que classe ele usa. A classe de um documento é sua principal guia de formatação/composição, sendo completada por pacotes, que não foram usados no exemplo mínimo.

Hello World!

Figura 3.1: Texto que será impresso em uma págica após o processamento do arquivo da listage 3.1

Cada classe pode possuir opções, como a opção `a4paper` usada no exemplo, e que define o tamanho do papel.

O símbolo de porcentagem indica o início de um comentário. Um arquivo LaTeXtem duas áreas principais, o preâmbulo, onde são colocados os comandos de configuração da saída, e o documento propriamente dito, onde é colocado o texto. O documento começa no comando `\begin{document}`. Tanto durante o preâmbulo quanto dentro documento podem aparecer comandos, mas eles tem objetivos diferentes.

O formato básico de um comando LaTeX é:

```
\comando[<opções>]{<parâmetro>}
```

porém existem comandos com colchetes após os parâmetros ou mais de um parâmetro, como veremos mais adiante.

Neste texto vamos usar a classe `article`, porém é muito comum que as pessoas usem outras classes, como classes fornecidas pela editora que vai publicar seu artigo ou livro. Classes bastante usadas são:

- `article`, genérica para artigos
- `report`, genérica para relatórios técnicos
- `book`, genérica para livro
- `IEEEtran`, classe para publicações diversas da IEEE

Uma das facilidades fornecidas por LaTeX é escrever seu artigo em uma classe genérica e depois simplesmente trocar a classe para a da editora desejada.

Existe uma classe para os documentos acadêmicos da Coppe, mantida por um grupo de voluntários que inclui este autor, e que cobre dissertação, tese, exame de qualificação e outras coisas. Ela pode ser obtida no GitHub no endereço `https://github.com/COPPE-UFRJ/CoppeTeX/tree/master/dist`

3.2 Estrutura de um documento

Um documento LaTeX utiliza uma estrutura de partes hieráquicas, que, nos principais estilos são:

- `\part{<título>}` – só para report e book
- `\chapter{<título>}` – só para report e book
- `\section{<título>}`
- `\subsection{<título>}`
- `\subsubsection{<título>}`
- `\paragraph{<texto>}`
- `\subparagraph{<texto>}`

A listagem 3.2 mostra um documento usando a classe `article` e divido em seções e subseções. **Parágrafos não marcados são separados por uma linha vazia** ou pelo comando `\par`. Uma prática comum em LaTeX é preencher um frase, da letra maiúscula ao ponto final, por linha de arquivo, e, obviamente, manter as linhas de um parágrafo juntas.

Apesar dos exemplos estarem gerando páginas de PDF, como mostrado na figura 3.2 a maior parte das figuras será cortada para evitar excesso de espaço em branco neste documento.

Listagem 3.2: Um artigo básico em LaTeX.

```
1 \documentclass{article}
2 % Introdução ao LaTeX
3 % Seminário LaTeX -- o Livro
4 % Geraldo Xexéo
5 % Este arquivo tem a licença Creative Commons
6 % BY-NC-SA 2020
7
8 \usepackage[T1]{fontenc}
9 \usepackage[english,brazilian]{babel}
10
11 \title{Meu Primeiro Artigo}
12 \author{Geraldo Xexéo}
13
14 \begin{document}
15
16 \maketitle
17
18 \section{A primeira seção}
19 Lorem ipsum dolor sit amet, consectetur
```

```
adipiscing elit. Duis ultricies efficitur aliquet.


\section{A segunda seção}
 Nulla iaculis bibendum sapien.
 Donec sollicitudin pharetra ipsum euismod vulputate.

\subsection{Uma subseção}
Sed ut nisl lectus.

Integer faucibus, est eu mattis vestibulum
\end{document}
```

3.3 Informação de Capa

Algumas informações que devem existir em um documento, que chamaremos de informação de capa ou de título, são colocadas no preâmbulo e inseridas automaticamente no documento por meio de comandos como **\maketitle** ou **\titlepage**.

Um artigo pode possuir um título, um subtítulo, um ou mais autores e uma data. A data, se não for colocada, é gerada automaticamente. A listagem 3.2 mostra os comandos **\title**, **\author**. O efeito deles acontece quando o comando **\maketitle** é chamado. O resultado é mostrado na figura 3.2. Outros estilos e pacotes permitem inserir a filiação do autor.

3.4 Comandos mais comuns

A figura 3.3 mostra alguns comandos mais comuns para formatar caracteres e seus resultados. Atenção ao uso das linhas vazias para manter ou trocar de parágrafo e ao fato que alguns comandos, como **\Large** tem efeito a partir do ponto em que ocorrem e tem que ser, de alguma forma, desfeitos, como foi feito com o comando **\normalsize**:

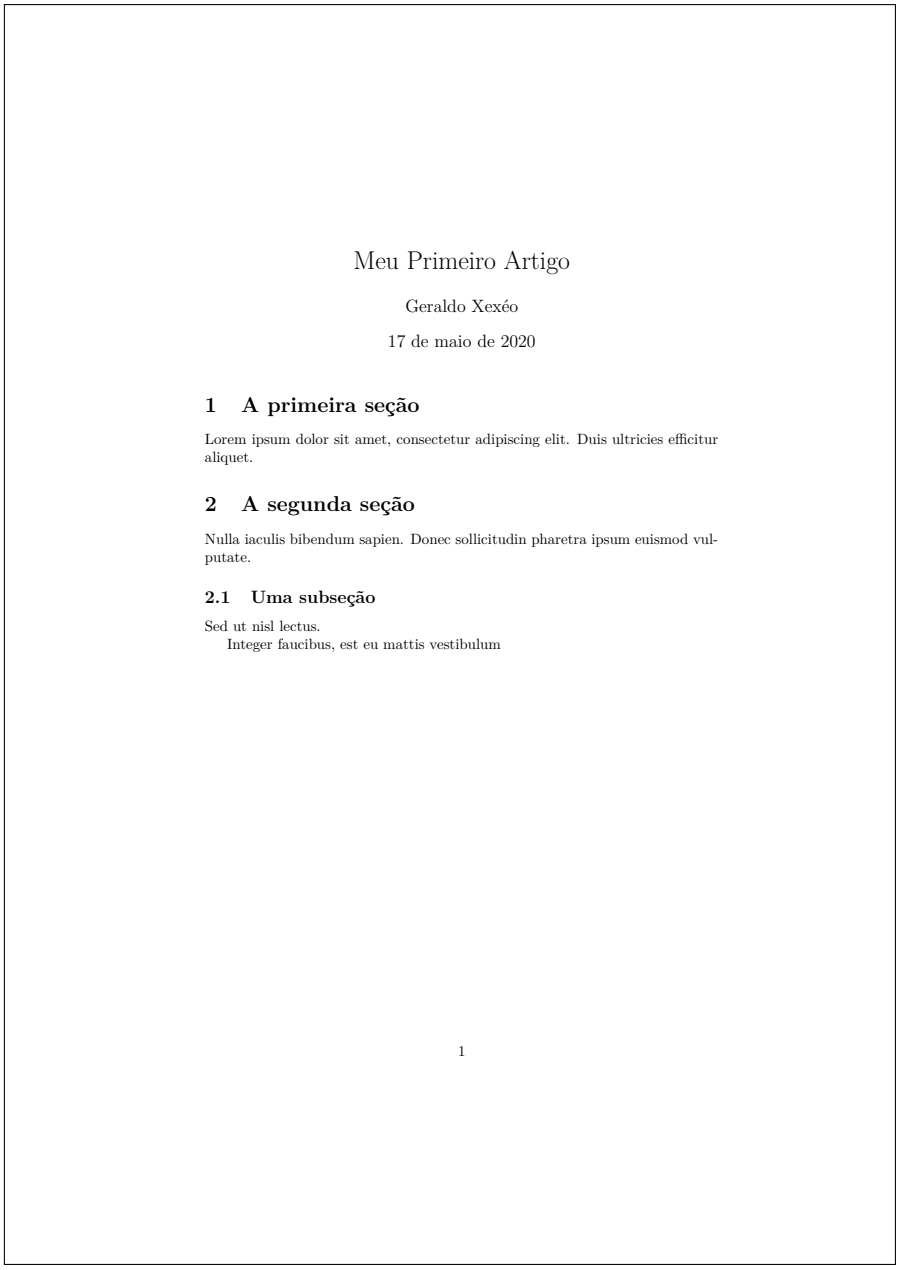


Figura 3.2: PDF produzido pelo texto da figura 3.2.

```
\textbf{negrito}
\textit{itálico}

\underline{sublinhado}

\Large
Texto Grande
\normalsize

e\textsuperscript{superescrito}
e\textsubscript{subescrito}
```

Figura 3.3: Comandos comuns em LaTeX

3.4.1 Fazendo referência a outra parte do documento

Muitas vezes em um documento é necessário citar alguma outra parte do mesmo, como uma figura, tabela ou seção. Para isso são usados dois comandos `\label{<rótulo>}`, que cria um rótulo para o local com o conhecimento do número a ser usado, como o número da figura, e `\ref{<rótulo>}`. Existem outras opções de citação específica como `\pageref{<rótulo>}`, que cita a página e não o contexto.

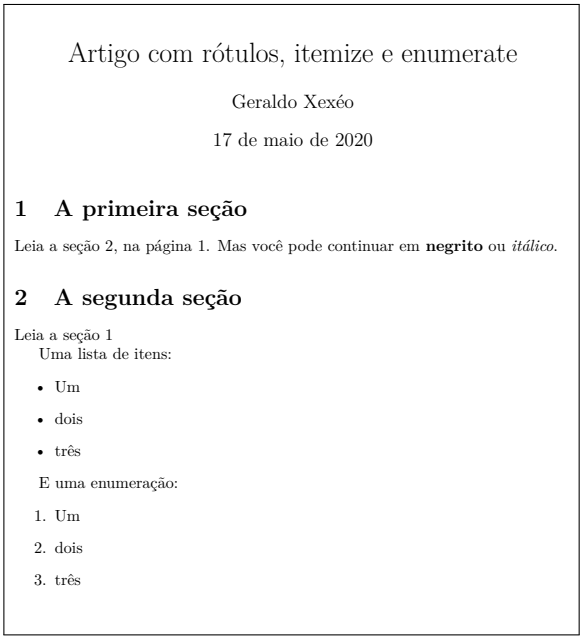
Listagem 3.3: Artigo usando as referências.

```
\documentclass{article}
% Introdução ao LaTeX
% Seminário LaTeX -- o Livro
% Geraldo Xexéo
% Este arquivo tem a licença Creative Commons
% BY-NC-SA 2020

\usepackage[T1]{fontenc}
\usepackage[english,brazilian]{babel}

```

```
\title{Artigo com rótulos, itemize e enumerate}

\author{Geraldo Xexéo}

\begin{document}

\maketitle

\section{A primeira seção}\label{sec:pri}
Leia a seção \ref{sec:seg}, na página \pageref{sec:seg}.
Mas você pode continuar em \textbf{negrito}
ou \textit{itálico}.

\section{A segunda seção}\label{sec:seg}
Leia a seção \ref{sec:pri}

Uma lista de itens:
\begin{itemize}
    \item Um
    \item dois
    \item três
\end{itemize}

E uma enumeração:
\begin{enumerate}
    \item Um
    \item dois
    \item três
\end{enumerate}

\end{document}
```

3.5 Usando pacotes

Pacotes são extensões a parte principal do LaTeXque aumentam seu poder de gerar saídas como desejado. Algumas extensões são tão importantes que são normalmente usadas, como o pacote Babel(Braams e Bezos, 2020), usado para escrever em outras línguas, dado o fato que LaTeXé originalmente criado para o inglês.


Artigo com rótulos, itemize e enumerate

Geraldo Xexéo

17 de maio de 2020

1 A primeira seção

Leia a seção 2, na página 1. Mas você pode continuar em **negrito** ou *itálico*.

2 A segunda seção

Leia a seção 1

Uma lista de itens:

- Um
- dois
- três

E uma enumeração:

1. Um
2. dois
3. três

Figura 3.4: PDF do exemplo do uso de referências (listagem 3.3).

O comando para usar pacotes tem o formato

`\usepackage[<opções>]{package}`.

Três pacotes essenciais para escrever em português do Brasil são:

- Babel
- inputencode
- fontencode

Ao procurar um pacote para resolver um problema é possível achar vários que façam a mesma coisa. Nesse caso é importante analisar se ainda estão sendo mantidos e qual sua aceitação na comunidade.

3.5.1 Babel

Babel(Braams e Bezos, 2020) é um pacote que permite usar o LaTeX com outras linguagens, já que ele é configurado naturalmente para o inglês. Isso implica em formas de separar palavras, palavras geradas automaticamente como *chapter* ou capítulo, e outras opções.

Os melhores pacotes usam o Babel para se autoconfigurar, quando imprimem algum texto automaticamente.

Um exemplo de comando para invocar o Babel é:

```
\usepackage[english,brazilian]{babel}
```

Ao contrário do que muitos usuários esperam, a linguagem mais importante é a última. Esse comando geralmente é um dos primeiros a ser dado, para que os outros pacotes se beneficiem dele, o que é comum.

A opção "portuguese" usa termos de Portugal, como dizer que um documento Web foi *acedido* em vez de *acessado*. Usar sempre **brazilian** no Brasil.

3.5.2 inputencode

- `\usepackage[utf8]{inputenc}`
- Faz o LaTeX entender código UTF-8 nos documentos que ele lê(Jeffrey e Mittelbach, 2013)
  - Caracteres acentuados do Português e outras línguas!
    - áéíóúâêîôûäëïöüàèìòù
- Você não precisa mais usar \t̂e
- **Não use o utf8x**, ele morreu e tem defeitos
- Deve ser o primeiro comando após a definição da classe do documento
- Melhor ainda, use o LuaLaTeX e dispense esse pacote!

3.5.3 fontencode

- `\usepackage[T1]{fontencode}`
- Quando gera o PDF, o LaTeX por *default* use o *font encoding* OT1, que só tem 7 bits.
  - Isso faz que uma leitura do texto do PDF para indexação, por exemplo, recupere combinações de caracteres que foram geradas para representar um caracter
    - Isso gera problemas na indexação do seu documento, ruim para você
- Garante também que as ligaduras, grande parte da beleza do texto do TeX sejam geradas, pois elas não são construídas, mas sim pertencentes as fontes.
- Sempre necessário

3.6 Ambientes

Ambientes são escopos fechados que são usados para mudar, temporariamente, o comportamento do LaTeX basicamente criando um contexto onde algo pode ser feito de acordo com regras específicas, como a criação de listas de itens, equações, figuras, etc. Em geral, um comando dado dentro do ambiente deixa de ser válido fora do ambiente.

Uma ambiente é marcado com um início e um fim, usando os comandos `\begin{<nome do ambiente>}` e `\end{<nome do ambiente>}`. Ambientes podem ser construídos um dentro do outro, de forma estruturada.

3.6.1 Ambiente Mais Usados

- `itemize` – ver listagem 3.3
- `enumerate` – ver listagem 3.3
- `equation`
- `tabular` – usado normalmente dentro de um ambiente table
- Floats – alguns ambiente "flutuam" no documento, sendo posicionados pelo algoritmo do LaTeX.
  - `figure`
  - `table`
  - `lstlisting` – depende do pacote `listings`

3.6.2 Equações

Você pode fazer equações dentro do texto, o que é conhecido no LaTeX como *inline*, ou em ambientes próprios, quando elas ficam isoladas, como mostrado na figura 3.5

Apesar de possuir bastante símbolos, existem pacotes da *American Mathematical Society* que adicionam várias capacidades mateméticas, como escrever equações em multi-linhas e muitos símbolos adicionais. Os dois principais pacotes são `amssymb` e `amsmath`.|

Construir equações em LaTeX é um aprendizado de uma sub-linguagem, que é, porém, bastante simples. Para iniciar esse aprendizado é interessante olhar sites que constroem equações interativamente, como `https://www.codecogs.com/latex/eqneditor.php`. Na verdade, basta buscar "latex equation online" no Google que encontrará rapidamente um site semelhante a imagem de figura 3.6.

```
Uma equação inline é construída a partir do caracter
\$, que no uso normal precisa ser feito com uma barra
antes, como em $e=\sum_{i=0}{n}1)/i!$.

Porém, para colocar a equação em destaque e poder citá-la
por uma referência numérica, como em equação \ref{eq:e},
deve ser usado o ambiente
 \lstinline|equation|.

\begin{equation}\label{eq:e}
e=\sum_{i=0}{n}1)/i!
\end{equation}

Se o número da equação não for desejado, o ambiente
correto é o \lstinline|equation*|, porém para isso é
 importante use o ótimo pacote
  \lstinline|\usepackage{amsmath}|.


\begin{equation*}
e=\sum_{i=0}{n}1)/i!
\end{equation*}
```

Uma equação inline é construída a partir do caracter $, que no uso normal precisa ser feito com uma barra antes, como em $e = \sum_{i=0} n1)/i!$.
Porém, para colocar a equação em destaque e poder citá-la por uma referência numérica, como em equação 3.1, deve ser usado o ambiente `equation`.

$$e = \sum_{i=0} n1)/i! \tag{3.1}$$

Se o número da equação não for desejado, o ambiente correto é o `equation*`, porém para isso é importante use o ótimo pacote `\usepackage{amsmath}`.

$$e = \sum_{i=0} n1)/i!$$

Figura 3.5: Exemplo de uso de equações.

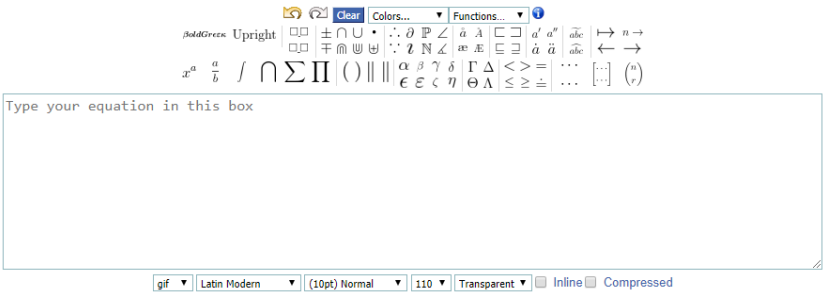


Figura 3.6: Exemplo de editor de equação on-line

3.7 Floats

Floats são ambientes que o LATEX posiciona no melhor lugar possível de acordo com suas regras de diagramação. O efeito de flutuação pode ser sentido neste documento, onde algumas vezes, para manter o fluxo de texto, as imagens migram mais para frente. Dois ambiente flutuantes são muito comuns: as figuras (`figure`) e as tabelas ()`table`).

É importante que alguns objetos flutuem porque eles não podem ser cortados ao meio, como uma figura. Então, a única coisa que o autor deve deve garantir que sejam menores que a página, enquanto o LATEX decide onde posicioná-los. Se forem maiores que uma página, há soluções para tabelas, e as figuras podem ser dimensionadas e cortadas com comandos específicos.

Flutuar significa que você não determina exatamente onde vão ficar, mas sugere ao algoritmo onde deseja colocar o ambiente flutuante. Isso se dá por meio de uma opção com letras ordenadas:

- h – here – tenta colocar na posição onde o comando está em relação ao texto;
- b – bottom – tenta colocar no fim da página, e
- t – top – tenta colocar no top da página.

Alguns autores recomendam não usar letra nenhuma e deixar o LATEX encontrar o melhor lugar. A figura 3.8 é um exemplo de figura que usa essa opção, já a figura 3.9 é um exemplo onde a opção não foi usada para uma tabela.

3.7.1 O Ambiente figure

Como o nome diz, o ambiente `figure` serve para inserir figuras em seu texto. Como a maior parte das pessoas faz figuras fora do LATEX, mesmo havendo

```
\begin{figure}[htb]
\centering
\includegraphics[height=0.3\textheight]{Images/Picture6}
\caption{Capa do livro de \LaTeX\ }
\label{fig:picture6}
\end{figure}
```

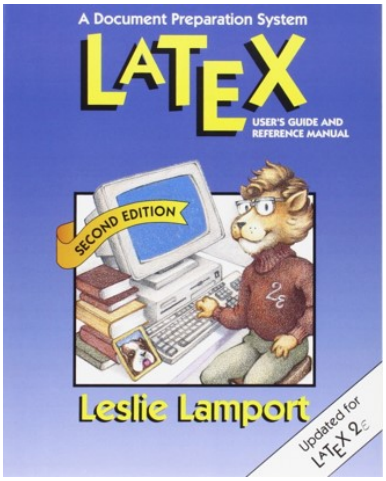


Figura 3.7: Capa do livro de LaTeX

Figura 3.8: Exemplo de uso do ambiente figure.

pacotes poderosíssimos de desenhos por comandos, ele é usado normalmente com o comando `\includegraphics[keyvals]{imagefile}`. Para isso é importante usar o pacote `graphicx`(D. P. Carlisle, 2017), que possui ainda comandos para fazer operações na figura, como cortar e colocar em escala, como fazemos com a opção `width` na 3.8. Também é possível cortá-las com as opções `trim` e `clip`.

3.7.2 Os Ambientes tabular e table

Para construir tabelas usamos o ambiente `tabular`, porém, para permitir que o LaTeX as coloque no lugar mais apropriado no texto, usamos o ambiente flutuante `table`.

O ambiente `table` é basicamente um *float* semelhante ao `figure`, as coisas

acontecem realmente no ambiente `tabular`. Aqui serão tratadas as opções para tabelas simples, mas existem comandos que permitem fazer qualquer tipo de tabela, como células agregadas que incorporam várias colunas ou linhas, por exemplo.

Todo `tabular` deve ser seguido da especificação das colunas da tabela. No exemplo da figura 3.9 isso é feito com a *string* `|c|c|`, que significa que haverá uma linha vertical para toda a tabela, uma célula com o conteúdo centralizado, outra linha vertical, outra célula centralizada e uma outra linha vertical. Já dentro do ambiente vemos dois tipos de linhas. Um tipo contém apenas o comando **`\hline`**, que cria uma linha horizontal, o outro contém dados, e dois símbolos são importantes: o `$` separa as colunas de uma linha, e o `\\` serve como separador de linhas.

Algumas das letras que definem o formato da tabela são:

- `l` – coluna alinhada a esquerda;
- `r` – coluna alinhada a direita;
- `c` – coluna centralizada;
- `|` – linha vertical do tamanho da tabela, e
- `p{wd}` – coluna onde cada item está em uma caixa de largura `wd`.

O ambiente `tabular` é adequado para texto, um ambiente semelhante, `array` é adequado para fórmulas. Existem pacotes que estendem as possibilidades de configuração de tabelas, como o `tabularx`(D. Carlisle, 2020b), `longtable`(D. Carlisle, 2020a), e outros. O pacote `array`(Mittelbach e D. Carlisle, 2020), por exemplo, é recomendado por trazer correções a configuração padrão. O pacote `booktabs`(Fear, 2020) também fornece opções extras de linhas que criam tabelas com melhor aparência, como foi usado na figura 3.10[1].

3.7.3 Ambientes para Listagens

Existem vários ambientes em LaTeX que permitem mostrar listagens de programas de computador e algoritmos. Para listagens de programas um bastante poderoso e configurável, e que está sendo usado neste texto para mostrar os arquivos em LaTeX, é o `lstlisting`(Heinz, Moses e Hoffmann, 2020), que faz parte do pacote `listings`[2].

Para algoritmos, existem várias opções viáveis, sendo que o mais atualizado é o `algorithm2e`(Fiorio, 2017).

[1] E ainda traz a sugestão de nunca usar linhas verticais ou linhas horizontais duplas.
[2] Atenção ao "s"

```
\begin{table}
\caption{Tabela de Idades}
\centering
\label{tab:idades}
\begin{tabular}{|c|c|}
\hline
\textbf{idade} & \textbf{nome} \\
\hline
0-2  & bebê \\
3-12 & criança \\
12-19 & adolescente \\
20-25 & jovem adulto \\
25-60 & adulto \\
60-80 & sênior \\
80-  & terceira idade \\
\hline
\end{tabular}
\end{table}
```

Tabela 3.1: Tabela de Idades

idade	**nome**
0-2	bebê
3-12	criança
12-19	adolescente
20-25	jovem adulto
25-60	adulto
60-80	sênior
80-	terceira idade

Figura 3.9: Exemplo de uso dos ambientes table e tabular.

```
\begin{table}
    \caption{Tabela de Idades}
    \centering
    \label{tab:idades}
    \begin{tabular}{cc}
        \toprule
        \textbf{idade} & \textbf{nome} \\
        \midrule
        0-2  & bebê \\
        3-12 & criança \\
        12-19 & adolescente \\
        20-25 & jovem adulto \\
        25-60 & adulto \\
        60-80 & sênior \\
        80-  & terceira idade \\
        \bottomrule
    \end{tabular}
\end{table}
```

Tabela 3.2: Tabela de Idades

idade	**nome**
0-2	bebê
3-12	criança
12-19	adolescente
20-25	jovem adulto
25-60	adulto
60-80	sênior
80-	terceira idade

Figura 3.10: Exemplo de uso dos ambientes `table` e `tabular` com comandos `booktabs`.

Capítulo 4

Controle de Referências

BibTeX é uma ferramenta e um formato de arquivos usados para descrever e processar listas de referência, normalmente em documentos LaTeX. Na prática, o termo é usado como termo geral para tratar dos arquivos de bases bibliográfica e de dois conjuntos distintos de aplicações associadas ao LaTeX:

- BibTeX(Patashnik, 1988) e biber(Kime e Charette, 2019), programas que processam a informação bibliográfica de um documento LaTeX e criam a bibliografia, também chamados de processadores de *backend*.
- natbib(Daly, 2010) e biblatex(Kime, Wemheuer e Lehman, 2019), pacotes LaTeX que formatam as citações e a bibliografia
  - natbib só funciona com BibTeX
  - biblatex funciona com ambos processadores

Apesar da maioria das pessoas usar um pacote de referências, como o natbib, o comando `\cite{}` é nativo do LaTeXe pode ser usado diretamente com o BibTeX sem nenhum pacote. Existe outro pacote menos conhecido mas usado na página do BibTeX que é o `cite`(Arseneau, 2015).

4.1 O arquivo .bib

O funcionamento básico do BibTeX depende de um arquivo .bib, exemplificado no figura 4.1, que é uma base de dados contendo referências contendo uma chave de citação. Essa base é construída em um arquivo texto comum, o que permite que seja manipulada tanto por programas específicos, quanto pelos autores diretamente. Ao longo do texto, o autor pode fazer citações a essas referências usando comandos como `\cite{Xexeo2020}`.

A interação entre o processador LaTeX, o pacote escolhido e o processador de referências escolhido, também conhecido com *backend engine*, trará para a versão a ser impressa as citações e a bibliografia no formato desejado.

Como pode ser visto na figura 4.1, um arquivo .bib é compostos de entradas, como `@book` que são, por sua vez, compostas de campos, obrigatórios ou opcionais, como `author`. As entradas geram formatos específicos na bibliografia, de acordo com as normas utilizadas.

Listagem 4.1: Exemplo de arquivo .bib

```
@book{sommerville:requirements,
author = {Sommerville, Ian and Sawyer, Pete},
title = {Requirements Engineering: A Good Practice Guide},
year = {1997},
isbn = {0471974447},
publisher = {John Wiley \& Sons, Inc.},
address = {USA},
edition = {1}
}

@article{therac25,
author = {Nancy G. Levenson and Clark S. Turner},
title = {An Investigation of the Therac-25 Accidentes},
journaltitle = {Computer},
date = {1993-07},
volume = {26},
number = {7},
pages = {18-41},
}
```

4.1.1 Tipos de Entradas Mais Usados

Os tipos de entradas mais usados, com seus campos obrigatórios, são(Kime, Wemheuer e Lehman, 2019):

- article – author, title, journal, year;
- inbook – author or editor, title, chapter and/or pages, publisher, year;
- book – author or editor, title, publisher, year;
- incollection – author, title, booktitle, publisher, year;
- collection – author, title, booktitle, year;
- inproceedings – author, title, booktitle, year/date

- proceedings –title, year/date;
- report – author, title, type, institution, year/date;
- thesis – author, title, type, institution, year/date;
- online – author/editor, title, year/date, doi/eprint/url, e
- misc – author/editor, title, year/date.

Deve se notar que os campos do biblatex são diferentes dos campos do natbib, por exemplo, o biblatex usa thesis onde o natbib usa phdthesis, porém o biblatex implementa todos os campos do natbib para compatibilidade com arquivos antigos.

4.2 O Ecossistema BIBTEX

A figura 4.1(ebosi e user2478, 2015), descreve o ecossistema BIBTEX. Ele é formado por pacotes que executam no processador LATEX, processadores para bibliografia, arquivos com bases de referência, programas que gerenciam esses arquivos (o que pode ser feito com editores de texto, no caso de arquivos .bib), e, possivelmente, outros programas que fazem transformações de arquivos. A figura mostra que há duas escolhas a serem feitas: o pacote LATEX a ser usado e o processador a ser escolhido.

Figura 4.1: O Ecossistema BIBTEX(ebosi e user2478, 2015).

A figura 4.2[1] mostra o funcionamento básico quando o biblatex e usado. Um ciclo que usa o natbib pode precisar de mais uma compilação com o

[1] É uma cópia da figura 2.2.

processador LaTeX de escolha. Basicamente o processador LaTeX gera um arquivo que contém as informações sobre as citações feitas no documento. Essa informação está no .bcf no caso do biblatex com o biber. Esse arquivo é usado, junto com os vários arquivos .bib contendo as bases de dados de referências utilizadas, para gerar um arquivo .bbl, que contém a bibliografia. Esse arquivo é usado, novamente pelo processador LaTeX, para gerar o documento final.

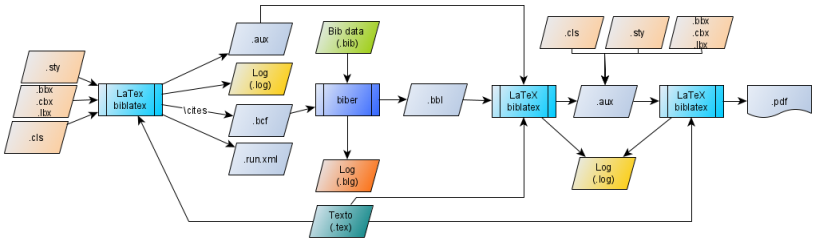


Figura 4.2: O fluxo de processamento (simplificado) do LaTeX, usando biblatex.

4.3 Comparando as Opções

Como se pode deduzir da figura 4.1, duas comparações básicas devem ser feitas: dos processadores, biber ou bibtex, e dos pacotes LaTeX, natbib ou biblatex.

4.3.1 natbib vs biblatex

As duas principais opções de pacotes LaTeX para referências são o natbib e o biblatex(Munn, 2015):

- **natbib** – é um pacote antigo e estável, porém ele é mantido mas não evoluído.
  - Vantagens
    - Vários formatos já definidos (arquivos .bst)
    - Possui um pacote custom-bib, com o aplicativo makebst, que gera estilos bibliográficos, .bst, iterativamente
  - Desvantagens
    - Depende do BibTeX, que tem algumas desvantagens.
    - .bst é difícil de fazer (linguagem posfixa)

        - ◇ Orientado para autor-ano e numérico, mas não autor-título, tipo de citação que aparece em outras áreas não tecnológicas
        - ◇ É fácil pegar a bibliografia gerada e incluir no arquivo .tex, o que algumas editoras pedem.
- **biblatex** – ativamente desenvolvido e em evolução, ligado ao *backend* mais poderoso biber. Substitui o natbib, e tem pacotes adicionais, como o `biblatex-abnt`.
    - Vantagens
        - ◇ Mais campos
        - ◇ Unicode no arquivo .bib, sem problemas para caracteres de outras línguas que podem aparecer em nomes de autores;
        - ◇ Usa métodos de LaTeX para controle fino da sua bibliografia;
    - Desvantagens
        - ◇ Algumas revistas podem não aceitar (porque não fizeram o *upgrade* de seus estilos).
        - ◇ Não é fácil pegar a bibliografia criada e inserir no arquivo .tex, o que algumas editoras pedem.

4.3.2 BibTeX vs biber

Os dois principais processadores de referências são o BibTeX e o biber, que podem ser comparados da seguinte forma(Munn, 2015):

- BibTeX
    - Use se for obrigado
    - Estável e debugado
    - Problemas com caracteres não UTF-8, exigindo o uso de códigos especiais
    - Funciona com natbib e biblatex
- biber
    - Sempre que possível, use
    - Suporta UTF8!
    - .bib file muito mais verificado e com regras mais fortes
        - ◇ Pode detectar um erro que não existia ou não era detectado em arquivos para o BibTeX
    - Só funciona com biblatex

4.4 Usando biblatex e biber

Sendo o par biblatex e biber formado pelos softwares mais modernos e em melhoria constante, eles compõe o par recomendado para o uso. São necessários alguns passos para isso ser feito:

1. ter um arquivo .bib com a base de citações;
2. usar os pacotes `csquotes`, se necessário escolhendo o estilo `brazilian`, e `xpatch`;
3. usar o pacote biblatex, escolhendo o *backend* `biber`, um estilo e possivelmente outras opções;
4. indicar os arquivos com as bases de referência que vão ser usados;
5. citar os documentos ao longo do artigo, e
6. incluir a bibliografia no texto.

Exemplos de comandos para fazer isso estão na listagem 4.2

Listagem 4.2: Exemplo de uso de biblatex

```
1 \usepackage[style=brazilian]{csquotes}
2 \usepackage{xpatch}
3 \usepackage[backend=biber,style=numeric]{biblatex}
4 \addbibresource{references.bib}
5 ...
6 \cite{biber:2012}
7 ...
8 \printbibliography
```

4.4.1 Exemplo dos comandos do biblatex

O biblatex permite várias formas diferentes de citação. A figura 4.3 mostra as principais, porém existem outras que podem ser encontradas no manual, (Kime, Wemheuer e Lehman, 2019), ou, para um guia rápido, (Rees, 2017).

Entre suas opções, o biblatex permite definir o estilo geral, definir o estilo da citação e da bibliografia separadamente, ordenar a bibliografia de várias formas e configurar as formas de citação para um formato específico. Também permite subdividir e bibliografia em partes diferentes a partir de chaves de seleção(Cassidy, 2013; Kime, Wemheuer e Lehman, 2019).

```
1
2 \autocite{biber:2012}
3
4 \cite{biber:2012}
5
6 \parencite{biber:2012}, ou mesmo
7  \parencite[veja][12]{biber:2012}
8
9 \textcite{biber:2012}
10
11 Lorem\footcite{biber:2012}
12
13 \fullcite{biber:2012}
```

(Munn, 2015)
biber:2012
(Munn, 2015), ou mesmo (veja Munn, 2015, p. 12)
Munn (2015)
Lorem
Alan Munn (15 de ago. de 2015). *bibtex vs. biber and biblatex vs. natbib. Some terminology*. TEX Stack Exchange. URL: https://tex.stackexchange.com/questions/25701/bibtex-vs-biber-and-biblatex-vs-natbib/299286 (acesso em 12/05/2020)

Figura 4.3: Exemplo do uso de formas de citações no biblatex

4.5 Programas Externos

Vários são os programas externos que podem ser usados para gerenciar arquivos .bib, entre eles:

- **JabRef** (Java) – um gerenciador de referências que usa o arquivo .bib como seu formato padrão e pode ser integrado com IDEs LaTeX;
- **Referencer** (GNOME) – outro gerenciador de referências, só funciona no GNOME;
- **BibTool** – A ferramenta de linha de comando para manipulação de arquivos BibTeX
- **BiB2x** – um processador que transforma do formato .bib para outros, e

- **Zotero mais Better BibTeX for Zotero** – uma extensão ao Zotero que permite gerar um arquivo .bib contendo as citações feitas em um documento .tex.

4.5.1 JabRef

O JabRef é um programa de gerenciamento de bases de referência que usa diretamente o formato le é escrito em Java, o que permite que rode em qualquer sistema operacional, sendo grátis e de código aberto, permitindo que sejam criadas extensões. A versão 5 lançou uma nova interface que pode ser vista na figura 4.4.

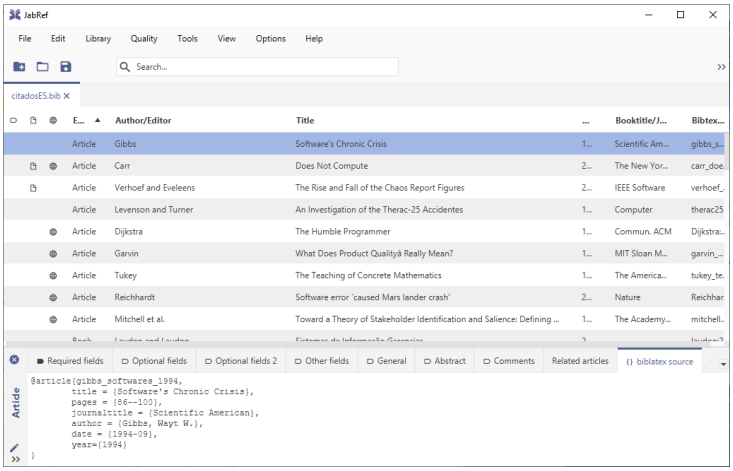


Figura 4.4: Exemplo de tela do Jabref.

Uma de suas características é permitir abrir vários arquivos e fazer manipulações entre os arquivos. Ele também suporta completamente o bibLaTeX, e é preciso configurar se o arquivo sendo tratado é para o natbib/bibtex ou para o BibLaTeX, por cada arquivo. Isso tem relação com os tipos de campos possíveis e com o uso da codificação UTF-8, por exemplo.

Ele é capaz de detectar erros nas entradas, sinalizando os campos com erro, e também por meio de um controle de qualidade que aparece no menu.

Uma funcionalidade pouco conhecida é a capacidade de enviar a citação direto para a IDE sendo usada, por meio de um botão (apontado na figura 4.5).

Figura 4.5: Botão no JabRef para inserir a citação atual no aplicativo de edição.

Capítulo 5

Para Onde Ir Agora?

Você está pronto a usar o LaTeX , escolha sua ferramenta e vá em frente. Aproveite.

Como conselhos finais:

- não tente mudar as coisas no início, comece pelo básico;
- comece na nuvem;
- faça parte da comunidade local, tire dúvidas com os amigos;
- google it!, escrever a pergunta certa é parte do entendimento;
- compartilhe as coisas interessantes que faz;
- aprenda um sistema de controle de versão (sim, faça backup), e
- **leia** outros livros sobre LaTeX, mas não deixe de ler o original (Lamport, 1994).

Bibliografia

Adams, Nico (mar. de 2007). *Polymer Informatics and The Semantic Web The Solution, Part 1: Adding Structure*. URL: https://semanticscience.wordpress.com/2007/03/30/polymer-informatics-and-the-semantic-web-the-solution-part-1-adding-structure/ (acesso em 03/2007).

Arseneau, Donald (27 de fev. de 2015). *The cite package:well formed numeric citations*. URL: http://linorg.usp.br/CTAN/macros/latex/contrib/cite/cite.pdf (acesso em 19/05/2020).

Braams, Johannes L. e Javier Bezos (13 de mai. de 2020). *Babel: Localization and internationalization Unicode TEX pdfTEX LuaTEX XETEX*. URL: http://linorg.usp.br/CTAN/macros/latex/required/babel/base/babel.pdf (acesso em 19/05/2020).

Carlisle, D. P. (1 de jun. de 2017). *Packages in the graphics bundle*. URL: http://linorg.usp.br/CTAN/macros/latex/required/graphics/grfguide.pdf (acesso em 21/05/2020).

Carlisle, David (7 de jan. de 2020a). *The longtable package*. URL: http://linorg.usp.br/CTAN/macros/latex/required/tools/longtable.pdf (acesso em 21/05/2020).

— (15 de jan. de 2020b). *The tabularx package*. URL: http://linorg.usp.br/CTAN/macros/latex/required/tools/tabularx.pdf (acesso em 21/05/2020).

Cassidy, Josh (13 de jul. de 2013). *Getting started with BibLaTeX*. URL: https://www.overleaf.com/learn/latex/Articles/Getting_started_with_BibLaTeX (acesso em 18/05/2020).

Daly, Patrick W. (13 de set. de 2010). *Natural Sciences Citations andReferences*. URL: http://linorg.usp.br/CTAN/macros/latex/contrib/natbib/natbib.pdf (acesso em 19/05/2020).

ebosi e user2478 (15 de ago. de 2015). *bibtex vs. biber and biblatex vs. natbib. answers to question*. TEX Stack Exchange. URL: https://tex.stackexchange.com/questions/25701/bibtex-vs-biber-and-biblatex-vs-natbib/299286 (acesso em 12/05/2020).

Fear, Simon (14 de jan. de 2020). *Publication quality tables in LaTeX*. URL: http://linorg.usp.br/CTAN/macros/latex/contrib/booktabs/booktabs.pdf (acesso em 21/05/2020).

Fiorio, Christophe (18 de jul. de 2017). *algorithm2e.sty package for algorithms*. Versão release 5.2. URL: http://linorg.usp.br/CTAN/macros/latex/contrib/algorithm2e/doc/algorithm2e.pdf (acesso em 21/05/2020).

Heinz, Carsten, Brooks Moses e Jobst Hoffmann (24 de mar. de 2020). *The `Listings` Package*. Versão 1.8d. URL: http://linorg.usp.br/CTAN/macros/latex/contrib/listings/listings.pdf (acesso em 21/05/2020).

Jeffrey, Alan e Frank Mittelbach (11 de ago. de 2013). *inputenc.sty*. URL: http://tug.ctan.org/tex-archive/macros/latex/base/inputenc.pdf (acesso em 19/05/2020).

Kime, Philip e François Charette (1 de dez. de 2019). *biber: A backend bibliography processor for biblatex*. Versão Version biber 2.14 (biblatex 3.14). URL: http://linorg.usp.br/CTAN/biblio/biber/documentation/biber.pdf (acesso em 19/05/2020).

Kime, Philip, Moritz Wemheuer e Philipp Lehman (1 de dez. de 2019). *The biblatex Package. Programmable Bibliographies and Citations*. Versão 3.14. URL: http://linorg.usp.br/CTAN/macros/latex/contrib/biblatex/doc/biblatex.pdf (acesso em 19/05/2020).

Lamport, Leslie (1994). *LaTeX(2nd Ed.): A Document Preparation System: Users Guide and Reference Manual*. USA: Addison-Wesley Longman Publishing Co., Inc. ISBN: 0201529831.

Mittelbach, Frank e David Carlisle (2 de fev. de 2020). *A new implementation of LaTeXs `tabular` and `array` environment*. URL: http://linorg.usp.br/CTAN/macros/latex/required/tools/array.pdf (acesso em 21/05/2020).

Mittelbach, Frank, Michel Goossens et al. (1999). *The LaTeX Companion (Tools and Techniques for Computer Typesetting)*. 2ª ed. USA: Addison-Wesley Longman Publishing Co., Inc. ISBN: 0201362996.

Munn, Alan (15 de ago. de 2015). *bibtex vs. biber and biblatex vs. natbib. Some terminology*. TeX Stack Exchange. URL: https://tex.stackexchange.com/questions/25701/bibtex-vs-biber-and-biblatex-vs-natbib/299286 (acesso em 12/05/2020).

Patashnik, Oren (8 de fev. de 1988). *BibTeXing*. URL: http://linorg.usp.br/CTAN/biblio/bibtex/base/btxdoc.pdf (acesso em 19/05/2020).

Rees, Clea F. (24 de jun. de 2017). *Biblatex Cheat Sheet*. URL: http://tug.ctan.org/info/biblatex-cheatsheet/biblatex-cheatsheet.pdf (acesso em 18/05/2020).

Reid, Brian K. (out. de 1980). “Scribe: A Document Specification Language and its Compiler”. Tese de dout. Department of Computer Science, Carnegie-Mellon University.